Mu2e-doc-5231-v2

Fermilab

Roadmap To Operations

Rob Kutschke

Mu2e Computing Review

March 6, 2015

Outline

- Draft milestones
- Housekeeping
- Strawman Org Chart
- Engaging non-experts
- Summary

External Constraints

- March 2016
  - DOE CD3c review
- Dec 1, 2015
  - Finish computing work for DOE CD3c review
- Q4 FY20
  - Start data taking with complete detector outside of the magnet.
  - This defines start of operations
- Goal:
  - Develop a plan to be ready for data taking in Q1 FY20
  - A major milestone every year or so.

Planning Milestones

- M1: April 1, 2015: Freeze code for CD3 production runs
- M2: Dec 1, 2015: Work for CD3 done
- M3: July 1, 2016
- M4: July 1, 2017
- M5: July 1, 2018
- M6: July 1, 2019
- M7: July 1 2020:
  - Cosmic Ray Tests; detector in the garage position.

M1 - April 1, 2015

- Next week: look at detailed tasks
  - Triage, prioritize and assign people
  - Highest priorities are:
    - Geometry updates
    - Changes to data products
    - Kill planes need for proton beam jobs, stage 1
    - Update mu2egrid for SAM and dCache
    - Finish commissioning: CVMFS, OSG running
    - Load test CVMFS, dCache OSG running
  - Many tasks are really reco or analysis phase projects and can be deferred until simulation production has started.
- Execute that plan

M2 Dec 1, 2015

- Execute the data processing campaign
- Analysis results ready for CD3c
  - Plus the development this implies
- Binary distribution and satellite releases working
- Proof of concept for running Offline trigger code in a simulated Trigger/DAQ environment:
  - Loop-free repository structure
  - Binary release of Offline visible to Trigger/DAQ
- Start to execute plan for engagement of non-experts.
- Expand validation suite
- Adopt coding standards document
- Start on the housekeeping list

M3 July 1, 2016

- Geometry and Conditions systems:
  - Reco geometry: nominal geometry plus conditions
  - Simulate with one geometry and reconstruct with a different one.
  - Can start alignment studies after this is in place.
  - Transition from text files to a conditions DB starting now.
  - Learn from LHC/BaBar etc
- Much improved event display; earlier if possible.
- Code reviews established
- Start to design data processing workflows; build and train the first group who will run them.

M4 – July 1, 2017

- Start date for:
  - Small scale data challenge
  - Small scale alignment challenge; not all degrees of freedom needed be covered.
- Conditions DB fully operational before this
- Demonstrate ability to reconstruct simulated cosmic ray tracks, field on and off.
  - Existing track finding code will not work at all
  - Track fitter will work.
  - Includes matching CRV with tracker and calorimeter
  - Understand value of field-off running for commissioning
  - Understand value of cosmic rays for alignment (field on & off)

M5 July 1, 2018

- Start date for a second iteration alignment challenge.
- Start date for a calibration challenges:
  - Momentum scale
  - Space time relation for straws
  - We will recognize more items as we approach this time

M6 July 1, 2019

- Start date for full scale data challenges:
  - Cosmic ray running
  - Data running
- Start date for full scale alignment challenge:
- Finish these by Jan 1, 2020 and we will have 6 months to address issues before cosmic ray data taking starts

M7 July 1, 2020

- Go!

Housekeeping

- Jobs that we need to do but don’t have firm deadlines
- Most are computing tasks but a few have physics content

Before Neutron/Cosmic Stage 2 or Beam Stage 4

- Define a data object that represents the energy deposition in one crystal in a more compact format than saving every G4Step and every G4Particle that contributes.
  - But keep enough information to have MC truth for cluster split-offs and albedo
  - Promises large reduction in disk space (>> 2x for files that have calorimeter info)
- We can move ahead without doing this but it will save storage space and transfer time.

Tracking

- Short term: Make persistent track data products
  - Data products are, by design, just data
  - Enough information to restore a fully functional track fit object that will give the same answer.
  - The big question is what functionality do we need to support for operations on persisted tracks without the need to restore the full track fit object.
    - Would like this to have the same public interface as a fully functional track fit object so that code is interchangeable.
- Longer term:
  - Modernize the BaBaR Kalman filter code
  - CLHEP -> Eigen within BaBaR code
    - ATLAS reports big speed improvements

Refactor Mu2e Offline Repository Structure

- Remove obsolete code and data products
  - Still available if you check out an old tag
- Refactor directory structure to break linkage loops
  - Start doing closed links
- Deploy BaBaR Kalman filter as an external product
- Split Mu2e Offline Repository
  - Core: all code needed for production, testing
  - One or more "analysis" repositories
    - Use the satellite release model to build these
    - Are allowed to be interdependent? (I vote no)
    - Will we allow data products to be defined in analysis repos?
    - Stntuple will move into one of these repositories

Code Housekeeping (1)

- Scrub code so that it compiles without warnings
  - Add –Werror to compiler flags
- Scrub code for names that were chosen on Opposite Day.
- Scrub misleading/redundant G4 and Mu2e prefixes from class/function/object names, directory names …
- Switch all enum-matched-to-string classes to the new style.
- Scrub magic numbers from production FHiCL files and replace with proper names.
  - Eg. 11 -> e_minus
  - Underlying support for this is already in place

Code Housekeeping (2)

- Tracker code was written before parts had established names. Class/object names do not match names on drawings and in documents.
  - Update it.
- Scrub code to eliminate unneeded headers and link libraries
  - And headers that should be in .cc not .hh
- Refactor geometry service to break compile time couplings.
- Scrub geometry *.txt files for unused/obsolete entries
- Scrub code to move inappropriate implementation from header to .cc

Code Housekeeping (3)

- When we have *art* with ROOT 6
  - Update persistency
- Recent FHiCL updates will allow us to rewrite top-level .fcl files in a way that is both more transparent and much more maintainable
  - @table, @sequence, @erase

Particle Data Table (PDT)

- Analyses need a PDT to interpret MC information
- Needs to G4-free but agree with G4 PDT
- Current use HepPDT. But …
- … HepPDT is broken.
  - Asked for fixes 5 years ago; no response.
  - It's small. Copy it and fix it.
  - Edit text table file to sync masses, names etc with G4 where appropriate
  - Instead of matching G4 names it might make more sense to make all names legal C++ identifiers so that they can be used as enum names?

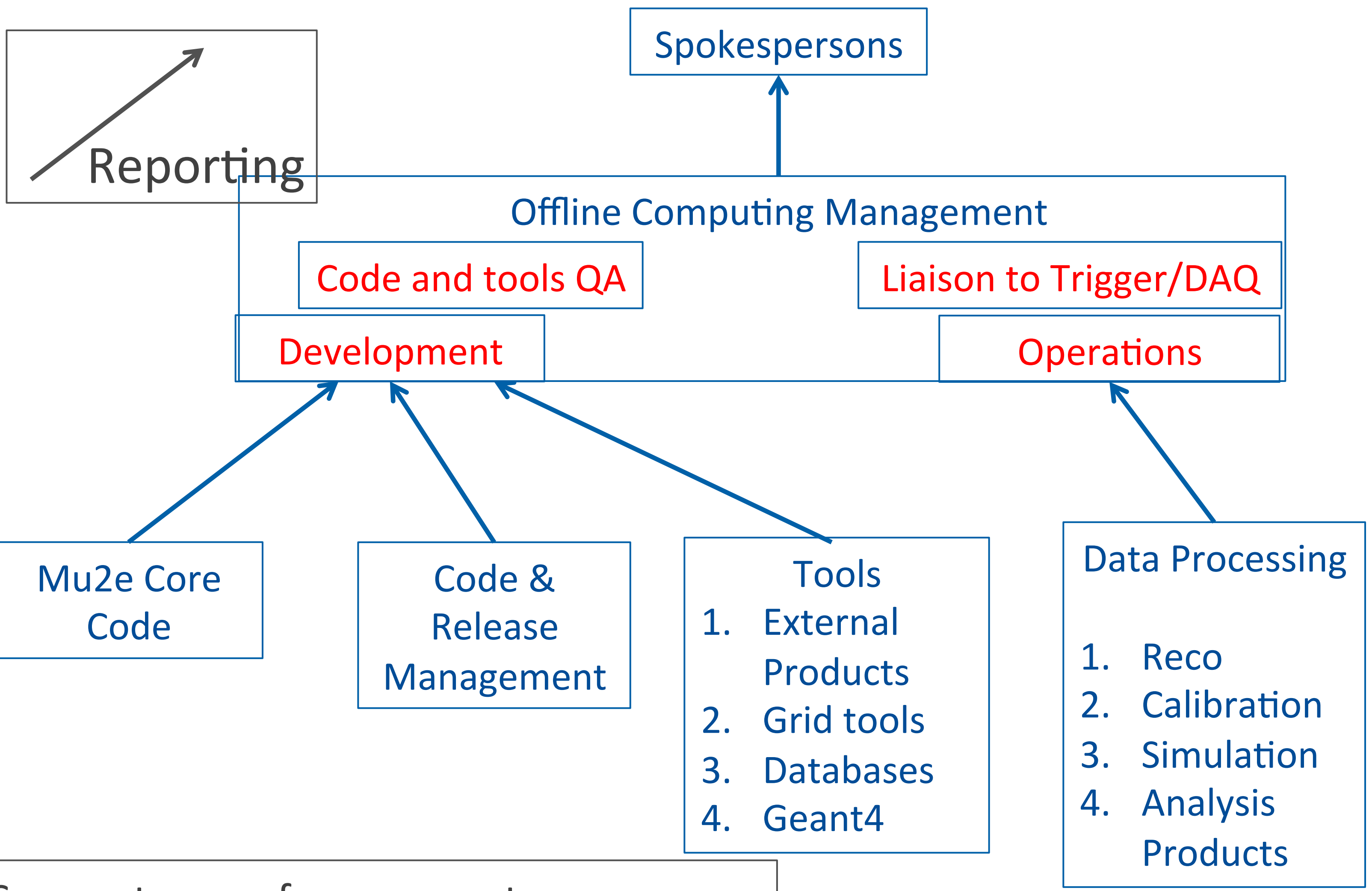
Code Management

- Present practice is push-to-remote-master
  - This works because we have a small community of developers with largely separate spheres of interest.
- Eventually want a request-to-pull model
  - Requires a dedicated code management staff
  - Pull code
  - Merge and validate the merged code
  - Reject code that fails validation or does not otherwise conform to standards
  - Add successful code to a release candidate branch.
  - Need to make sure that this does not become a bottleneck
  - The payoff is a more robust code suite

Management Structure for Operations

- Operations is in 2020 so we have not spent much time on this
- At this stage all that we can really do is list the roles that need to be filled.
- The next page is a strawman for an org chart that is based on an old CDF org chart plus some perturbations.
- There are still a few glitches in this model and we will look at other models over the next few years.

Strawman Management Structure during Operations?



See next pages for comments

Comments on the Strawman Org Chart

- Offline Computing Management has 2 groups of functions:
  - A development and operational
  - Two things don't fit nicely into this breakdown
    - A QA organization
    - A liaison to the trigger/daq/online organization(s)
- Under "Tools" and "Data Processing" the numbered bullets are subgroups.
- Where do algorithms and calibration codes live?
  - Probably in the appropriate detector or analysis group.
  - Need to discuss with stakeholders
- Core code is anything that Mu2e maintains that is not an algorithm.

Comments on the Strawman Org Chart

- QA means QA for code, scripts, operational procedures.
  - Needs to been near the top of the organization
- Code managements
  - Use a request-to-pull model
  - QA gets involved here
  - Validate and merge into a release candidate branch
- Release management
  - Work with stakeholders to decide what functionality belongs in which release.
  - Incorporates validation code provided by algorithm groups and others.
  - Runs validation code used to validate releases.

Engaging non-Experts

- It's on our radar now and we have ideas
  - How do we onboard new people?
    - Those who will work with Mu2e Offline
    - Those who will work with "ntuples"
- Other people may have different ideas
- Next step: work with collaboration to develop a phased plan.
- Start to execute the plan by fall 2015
- How fast we work through the plan depends on available resources and competing priorities.
  - Collaboration needs to be involved in setting scope and priorities
- Must avoid the first-out-of-the-gate wins problem.
  - We must adopt a robust long term solution.

Summary

- We have presented a rough draft of how we get from now to the start of operations
  - In the short term we have a lot of detail.
  - In the longer term we just have highlights and milestones
- There is a large body of housekeeping work
  - This could greatly benefit from a utility programmer
- We have a strawman for the computing org chart in 2020
  - We have a list of questions to ask
- Engaging non-experts is a priority and we will consult with the collaboration this summer to develop a phased plan.